F. Chlebana
Feb 14 2007

CSL Operations Meeting

Status

Monitoring

Additional information collected at:

http://ncdf76.fnal.gov/ chlebana/daq/cslUpgrade/monitor/

Code Updates

Status

We did have two cases of truncated db files...

```
Tue Feb 13 10:00:01 CST 2007
  Found corrupted db file in b0csl22
ar0391c9.010dphys.db

Tue Feb 13 10:00:01 CST 2007
  Found corrupted db file in b0csl28
hr0391c9.0101phys.db
```

We should add the file time stamp to the “Truncated db file recovery status”

Otherwise working well...

→ Most major operational issues are resolved?

→ Focus on robustness and monitoring

Review Functional Requirements for Monitoring

→ Inform shift crew of CSL related problems

- *Message sent to Error Logger*
- Should be simple: *Identify CSL as the problem source*

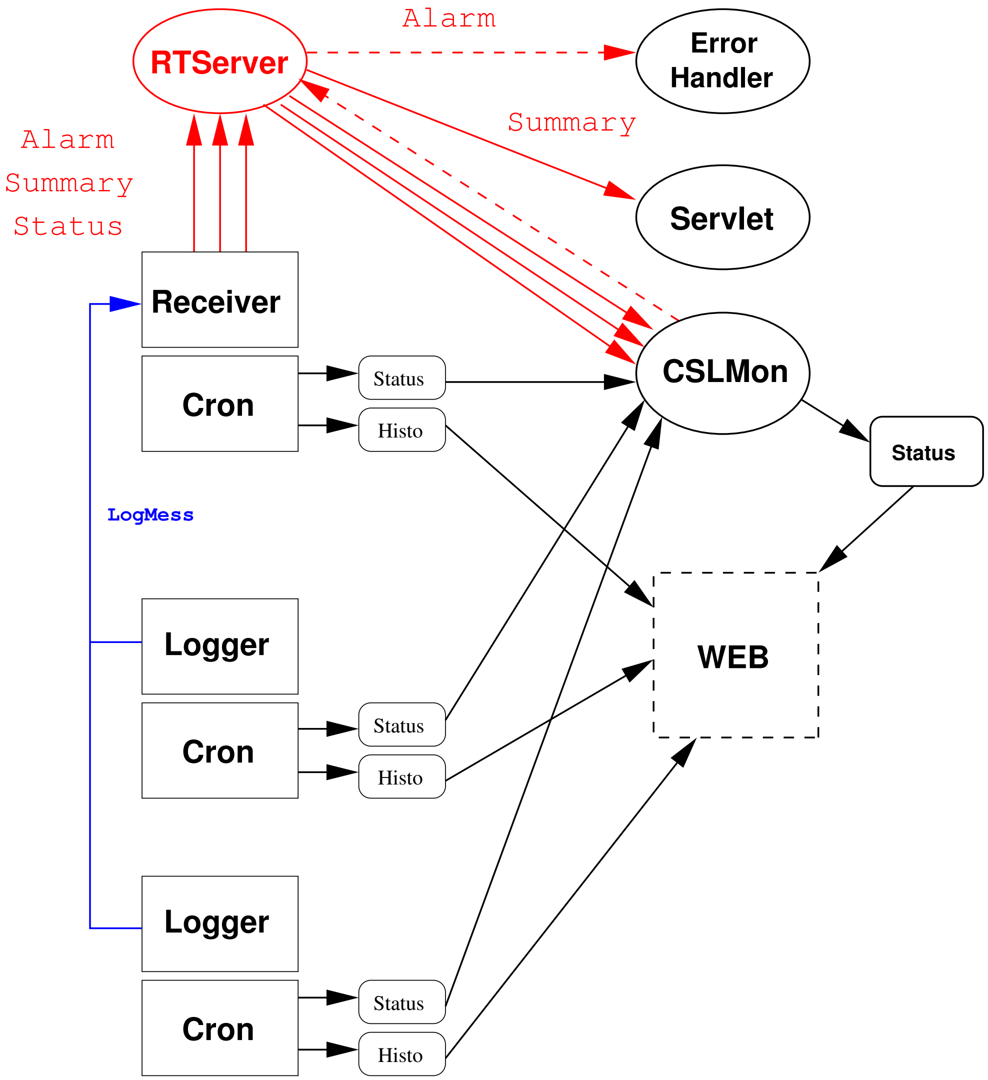
→ Guide shift crew through recovery procedure

- *Error handler displays recovery procedure*

```
Run Stopped due to CSL problem.
End current run, restart the CSL and start new run
```

→ Additional information for experts to be displayed on CSLMon.

- *More detailed information displayed by CSLMon*
- *History of messages*
- *Log files*



CSLMon receives information from various sources

→ *Receiver*

→ *Logger Status Files*

CSLMon decides if the shift crew should be notified based on this information

Can change decision algorithm without requiring a new version of the CSL code.

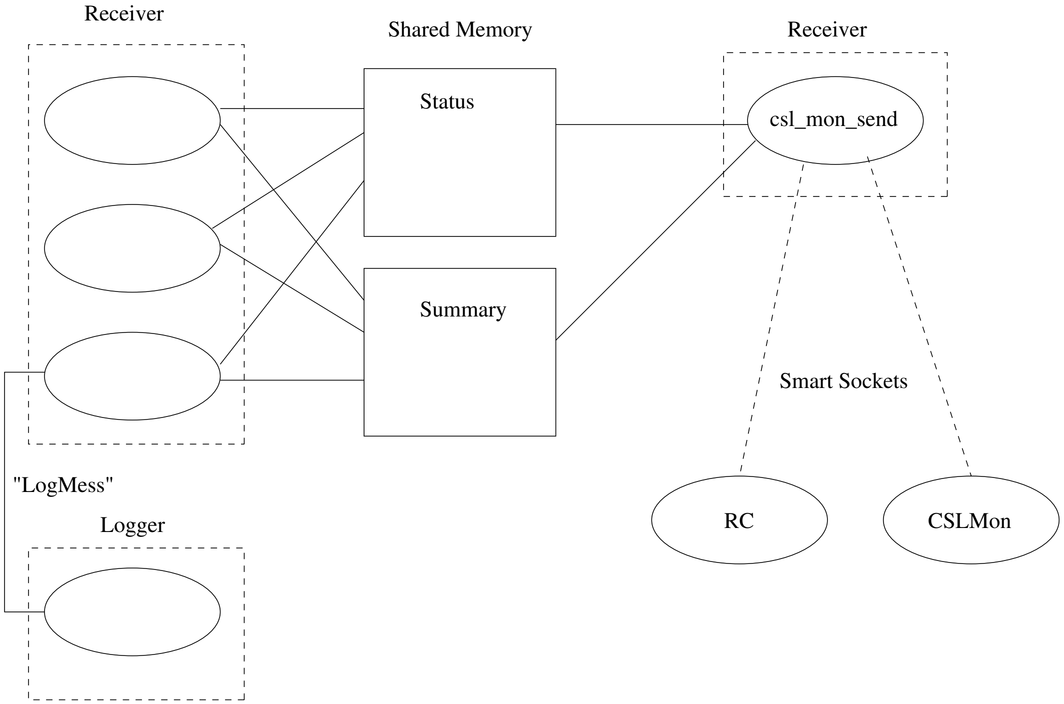
Several people working on this and we should clarify who is doing what and make sure we all agree on what we want to accomplish

Message Flow

- Receiver sends CSL Status, CSL Summary
- Loggers write status files
- CSLMon receives messages and reads files

→ *Determines if there is a problem and sends Alarm to Error handler*

→ All Alarms originate from CSLMon



Need to clarify the content of the messages

Need to clarify at what point the information is written out

Merlin Smart Sockets CSL Messages

We have *many* messages that are defined as Merlin messages

```
* Mnemonic: CSL_RECEIVER_MSG_TRUNCATION; Code: 0xc230800000041
  Text: Event truncation detected in partition %d. Maybe event size is too large.

* Mnemonic: CSL_RUN_BAD_EVENT_SUMMARY; Code: 0xc220200444445
  Text: For partition %d, run %d, \n there were %d bad BORs, %d bad Events, %d bad
```

Sent out as smart socket messages to the Error Handler (From the Receiver)...

Are they still used?

→ *Do not think that they reflect the type of problems we have with the new system.*

→ *Too much detail to be sent to the Error Handler*

Review error conditions, assign severity and define recovery procedure

Error Condition

```
1) CSL_NEVENTSERROR > 2
2) CSL_NEVENTSERROR increasing
3) Unable to Log data
```

Error Level (Warning/Error/Severe)

```
1) Warning
2) Error
3) Error
```

Recovery Procedure/Message to Error Handler

```
1) ‘‘Level 3 Filter Error’’
2) ‘‘High rate of Bad events from L3’’
3) ‘‘Unable to Log data, Stop current Run
   Restart CSL and start new run’’
```

CSLMon

We should already have enough information to recognize when we have bad events.

1) Make CSLMon aware of this error

2) Have CSLMon send Alarm to Error Handler

3) When CSLMon starts can is display a history of messages

Code Updates

1) Make sure that the file boundary/file manifest is working

- Improve splitting of files
- Make sure all files are added to the manifest

2) Make sure that the logger switching problem is fixed

(Quietly throwing away event problem)

- CSL should block if it cannot write out events

3) Verify Bad Event Handling

- Bad events with no stream information should be written to the local area on the reciever

- Bad events with stream information should be sent to the logger and written to the data stream. They should not be written to the error directory on the logger node. They should be written to the error directory on the receiver node

4) Implement new error messages

- Error count > 2
- Error count increasing
- Error message if logger cannot write out data

Information already be available to CSLMon through the Status/Summary message.

Are there any error conditions that we should be aware of that cannot be recognized from the information already available?

5) Test robustness of restart scripts

- Restart in the middle of a run
- Test CSL/L3 connection
  We should not have to clean up L3 when the CSL is restarted
- Add CSL restart to ProcMon

6) Make sure Calibration CSL is independent of main CSL

- Change smart sockets subject